UNITED STATES PATENT APPLICATION

For

LOOK-UP TABLE FOR TRANSFER FUNCTION

Inventors:

Bradley C. Aldrich
Moinul H. Khan
Kayla Chalmers

Prepared by:
Blakely, Sokoloff, Taylor & Zafman
12400 Wilshire Boulevard
Seventh Floor
Los Angeles, California 90025
(408) 947-8200

Attorney's Docket No. 42390P19127

"Express Mail" mailing label number: EV339914492US
Date of Deposit: March 29, 2004
I hereby certify that I am causing this paper or fee to be deposited with the United States Postal Service "Express Mail Post Office to Addressee" service on the date indicated above and that this paper or fee has been addressed to the Commissioner for Patents, P.O. Box 1450, Alexandria, VA 22313-1450.

Vineta T. Tufono
(Typed or printed name of person mailing paper or fee)

Vineta T. Tufono
(Signature of person mailing paper or fee)

3-29-2004
(Date signed)

LOOK-UP TABLE FOR TRANSFER FUNCTION

FIELD OF THE INVENTION

[0001] Embodiments of the present invention relate generally to image data processing, and in particular, look-up tables used in image data processing.

BACKGROUND

[0002] Today, an ever-increasing number of mobile devices, such as PDAs and cellular telephones, are being outfitted with digital cameras and video recorders. Such digital cameras use sensors to which capture images in a digital data format, i.e., as bits; zeros and ones. Digital data processing involves manipulating these bits to enhance the resultant image, or for other purposes, such as convenient storage.

[0003] There are numerous well-known image data processing operations that can be performed on digital image data. These include companding (reducing the bit-depth of the image data), gamma correction (compensating for brightness differences), sensor correction (compensating for sensor non-linearity), and illumination correction (compensating for changed illumination environment, e.g. inside or outside) to name a few. All of the data processing operations mentioned above may be implemented using a transfer function to describe the appropriate adjustment. A transfer function is merely a function that maps input values to output values according to a formula or curve.

[0004] Transfer functions can be implemented in hardware as look-up tables (LUT) to increase processing speed. However, a full LUT representing all possible

outputs of a transfer function takes up a lot of memory. One way to reduce the memory required to store the LUT is to have the LUT store only sample outputs of the transfer function. However, sampling the transfer function compromises accuracy and flexibility.

BRIEF DESCRIPTION OF THE DRAWINGS

[0005] Embodiments of the present invention are illustrated by way of example, and not by way of limitation, in the figures of the accompanying drawings in which like reference numerals refer to similar elements and in which:

[0006] **Figure 1** is a block diagram of an image processing device according to one embodiment of the present invention;

[0007] **Figure 2** is a block diagram of a transfer function module according to one embodiment of the present invention;

[0008] **Figure 3** is a graph demonstrating two example transfer functions;

[0009] **Figure 4** is a graph demonstrating partitioning a transfer function according to one embodiment of the present invention;

[0010] **Figure 5** is a flow diagram of reprogramming a look-up table according to one embodiment of the present invention; and

[0011] **Figure 6** is an architectural block diagram demonstrating image processing by an image processing device according to one embodiment of the present invention.

DETAILED DESCRIPTION

Image Processing Device

[0012] Figure 1 shows an example generic image processing device (IPD) 100 in which embodiments of the present invention can be implemented. The IPD can be a digital camera (either one capable of only still photography, one capable of video capture, also known as a camcorder, or both), a cellular phone with imaging capabilities, a personal digital assistant (PDA) with imaging capabilities, a personal computer (PC), a mobile computer such as a laptop, or any other computing device with image processing capabilities. The IPD 100 need not poses image capturing capabilities, the image data can be fed from some source to the IPD 100.

[0013] In one embodiment, the IPD 100 includes a sensor 102 that can convert light into digital image data. The sensor 102 can be a Complementary Metal Oxide Semiconductor (CMOS) image sensor, a Charge Coupled Device (CCD) - type sensor, or any other sensor used for digital photography. The IPD 100 may further include any number of image data sources 104, such as a processor or some memory that feeds image data onto the image data bus 106. The image data bus 106 can carry image data output by the sensor 102 or other image data source 104, in various formats, such as raw image data of any depth (e.g., 12 bit, 10 bit, 9 bit, 8 bit) or color space-specific pre-processed data, e.g., YcbCr or RGB data (8 bit).

[0014] The image data can then undergo various image data processing. In one embodiment, if the image data is in a 9 or 10 bit format, the image data can be companded into an 8 bit format to increase processing speed, reduce the storage requirements, and decrease the memory bandwidth required to transfer the image to the

targeted memory resources. A companding (compress-expand) operation can be performed using a transfer function. A transfer function maps inputs in the input range to outputs in the output range and describes the functioning of the appropriate system, in this case, the companding operation. The transfer function can be linear, but more often is represented by a curve.

[0015] To increase processing speed, a transfer function can be represented by look-up table (LUT). A LUT can be an indexed data structure storing the outputs of the transfer function. The mapping of the transfer function can then be represented by an addressing scheme that maps the inputs to an index of the LUT, where the indexed output represents the output associated with the input by the transfer function.

[0016] Transfer functions can also be used for gamma correction, compensating for sensor non-linearity, illumination correction, and various other image processing operations. In one embodiment, all required corrections can be implemented using a single transfer function. Thus, the image data from the image data bus 106 is further processed by the transfer function module (TFM) 108, which maps the input image data to output image data according to a transfer function. Since transfer function can be used for various data processing operations, the TFM 108 has a generic name. However, in some embodiments, the TFM 108 could be referred to as a compand module, gamma correction module, and so on, according to specific functionality associated with the transfer function.

[0017] In one embodiment, the image data is also observed by a histogram module 110. The histogram module can tag colors and perform statistical analysis on the incoming image data to determine the appropriate transfer function to be used for the

appropriate operations. Thus, in one embodiment, the histogram module 100 provides the information used to program the LUT(s) implementing the appropriate transfer function. How the appropriate transfer function is determined by the histogram module 110 is well understood by those skilled in the art.

[0018] The IPD 100 can include various other components. In one embodiment, the IPD 100 includes a processor 112 controlling the operation of the IPD 100, and a memory 114 to store an operating system and for storage to store the output of the TFM 108. The IPD 100 can also have further processing modules 116 that process the image data further. The IPD 100 can also have a display 118 that can display the image data as an image to a user. The IPD 100 can also have various other components, such as a user interface, a radio transceiver, an audio input/output, and other hardware and software depending on the specific type of IPD 100 implemented.

Transfer Functions Module

[0019] One embodiment of the present invention is now described with reference to Figure 2. Figure 2 shows an expanded view of the TFM 108 from Figure 1. The input of the TFM 108 is image data in any format, and the output of the TFM 108 is here generally referred to as transferred image data. This generic term is used, since in some embodiments, the output may be gamma corrected image data, companded image data, and so on.

[0020] In one embodiment, the image data is first received by a color channel filter (CCF) 120. The CCF 120 determines the color of the input data based on the image protocol being used. For example, for raw Bayer pattern RGB image data, the CCF 120 would know to expect red/green alternating for one image row, and green/blue alternating

for the next image row. In one embodiment, the color of the input image data determines which LUT 124 to use. In Figure 2, there are three LUTs 124 shown for simplicity. The specific number of the LUTs depends, in one embodiment, on the number of colors (or other indicators such as hue or brightness) used by the color space of the image data.

[0021] In one embodiment, the image data next enters the address module 122. The address module 122 maps the image data to an index into the selected LUT 124. In other words, the image data is used to address into the LUT 124. The index, i.e., address, is mapped such that the LUT entry at the address is the output of the transfer function associated with the input image data.

[0022] As mentioned above, in one embodiment, the LUT 124 does not contain all possible outputs in the output range of the transfer function. Instead, the transfer function is sampled. That is, the input range of the transfer function is divided into segments, and one sample input is chosen for each segment. For example, a sample input is chosen at every 6^{th} possible input over the input range. Then, the transfer function is calculated for these sample inputs, and these sample outputs can be stored in the LUT 124. Some accuracy lost by sampling can be regained by passing sample output from the LUT 124 to an interpolation module 126 where some kind of interpolation can be performed to create an output value between two sample output values. Various interpolation techniques, including linear interpolation can be implemented by the interpolation module 126.

Look-up Tables

[0023] As discussed above, transfer functions are generally represented by a curve that maps the input range to the output range. When the curve is close to a straight

line, i.e., has relatively low curvature, much of the accuracy lost by sampling can be regained using interpolation. However, when the curve is not straight or close to straight, i.e., has high curvature, interpolation cannot recapture some, or much, of the accuracy lost to sampling. The difference between high or low curvature can be dependent on error toleration factors, and can be expressed mathematically by the absolute value of the second order derivative of the transfer function. In one embodiment, a high curvature region is defined as one with an average curvature above a certain threshold, e.g., 5.

[0024] Many real life transfer functions have both high-curvature and low-curvature portions, as shown in Figure 3. Since LUTs 124 are fixed in size, they can only store a certain number of samples. Since even a relatively few samples can accurately reflect the low-curvature regions of the transfer function, when used with interpolation, it can help accuracy to reallocate more of the fixed number of samples to the high-curvature regions of the transfer function. Thus, in one embodiment, the LUT 124 is re-programmable by the histogram module 110 to flexibly allocate more output samples to high-curvature regions. In one embodiment, the addressing scheme performed by the address module 122 takes account of this flexibility and properly maps the input image data to LUT addresses despite the flexibility of the LUT 124.

[0025] In one embodiment, the transfer function to be represented by the LUT 124 is divided into equal, or substantially equal regions over the input range. An embodiment using four quartile regions (or quartiles) is shown in Figure 4. Quartiles Q3 and Q4 are low curvature regions in which a low number of samples may be sufficient. However, quartiles Q1 and Q2 are higher-curvature regions, where accuracy can be

significantly improved by using a more samples. The embodiments of the present invention are not limited to four such regions, more or less can be used as desired.

[0026] One embodiment of re-programming the LUT 124 is now described with reference to Figure 5. In block 502, a transfer function to be used for processing image data is generated. In one embodiment, this can be done according to known statistical methods. In block 504, the input range of the transfer function is partitioned into regions. In one embodiment, the regions are substantially equal in size. There can be any number of regions. In three example embodiments there are two, four, and eight regions respectively.

[0027] In block 506, a determination is made as to whether any of the regions are high-curvature regions, as explained with reference to Figures 3 and 4. In there are no high-curvature regions, the sample inputs are allocated uniformly across the various regions, in block 510. However, if one or more regions are identified as high-curvature, then, in block 508, more sample inputs are allocated to these high-curvature regions. In one embodiment, up to half (50 percent) of the regions can be designated as high-curvature.

[0028] In one embodiment the number of input samples equals the number of possible entries in the LUT 124. After all the input samples are allocated across the regions according to either block 508 or 510, the transfer function is applied to the input samples, and the results (output samples) are used to populate the entries of the LUT 124.

Demonstrative Example

[0029] One embodiment for a hardware implementation of the TFM 108 is now described with reference to Figure 6. In Figure 6, the image data from the image data bus

106 first passes through the CCF 120. As described above, the CCF 120 determines the color of the image data and selects the appropriate LUT 124 based on this color. In this embodiment, the LUTs 124 are 48X8 bit tables, one for each color channel. Each 8-bit entry is an output sample of a transfer function implemented by the TFM 108.

[0030] In the embodiment shown in Figure 6, the CCF 120 also determines which quadrant the input image data belongs to. Depending on the quadrant, the CCF 120 divides the image data up into three fields, here referred descriptively as Q (quadrant), C (coarseness) and R (residue) fields. Non-descriptivelly, these fields can be referred to as first, second, and third parts or sections of the image data, respectively.

[0031] Since there are four quadrants, two bits will identify the quadrant of the image data. Thus, the most significant two bits make up the Q field. The number of bits in the C field (here referred to as L_{Qi}) depends on the quadrant, and more specifically on how many samples were allocated to the particular quadrant. For example, if the input image data belongs to a quartile with 4 samples, the C field can have 2 bits. However, if the quartile has 8 samples, the C filed can have 3 bits. The bits not allocated to either Q or C fields are field R, the residual bits. The size of the R field depends on the size of the C field and the number of input bits in the image data.

[0032] Multiplexer 128 uses the Q field to select the appropriate quartile pointer 130, as indicated by the Q field. Upon the programming of the LUTs 124, when the number of samples per quartile is set, the quartile pointers 130 are calculated, in one embodiment, as follows in Equations 1-4:

$$\text{Q1-Pointer} = \text{base address of channel LUT} \quad \text{(Eq. 1)}$$
$$\text{Q2-Pointer} = \text{Q1-Pointer} + 2L_{Q1} \quad \text{(Eq. 2)}$$
$$\text{Q3-Pointer} = \text{Q1-Pointer} + 2L_{Q1} + 2L_{Q2} \quad \text{(Eq. 3)}$$
$$\text{Q4-Pointer} = \text{Q1-Pointer} + 2L_{Q1} + 2L_{Q2} + 2L_{Q3} \quad \text{(Eq. 4)}$$

[0033] where L_{Qi} is the number of samples allocated to quartile number i.

[0034] Thus, if the Q field places the input data image in quartile Q2, then multiplexer 128 selects the Q2-Pointer from the quartile pointers 130. In one embodiment, multiplexer 132 selects the C field of the image data and merge 134 merges the selected quartile pointer with the C field to generate the index of the LUT entry. That is, the quartile pointer and the C field are combined to address into the selected LUT 124. The entry corresponding the this index is referred to as E(i) in Figure 6.

[0035] Next, interpolation is performed to produce the transferred image data. In one embodiment, the interpolation produces the result shown in Equation 5:

$$Out(i) = E(i) + \frac{E(i+1) - E(i)}{2^r} \cdot R \qquad \text{(Eq. 5)}$$

[0036] where Out(i) is the output of the TFM 108, i.e., the transferred image data, E(i+1) is the LUT entry after the indexed LUT entry, R is the binary value in the R field of the image data, and r is the number of bits in the R field. In one embodiment, arithmetic logic units 136 and 142 are used in conjunction with multiplier 138 and shift register 140 in the arrangement shown in Figure 6 to perform the calculations required by Equation 5.

General Matters

[0037] Although the present system will be discussed with reference to various illustrated examples, these examples should not be read to limit the broader spirit and scope of the embodiments of the present invention. Some portions of the detailed description that follows are presented in terms of algorithms and symbolic

representations of operations on data within a computer memory. These algorithmic descriptions and representations are the means used by those skilled in the computer science arts to most effectively convey the substance of their work to others skilled in the art. An algorithm is here, and generally, conceived to be a self-consistent sequence of operations or acts leading to a desired result. Usually, though not necessarily, these quantities take the form of electrical or magnetic signals capable of being stored, transferred, combined, compared and otherwise manipulated.

[0038] It has proven convenient at times, principally for reasons of common usage, to refer to these signals as bits, values, elements, symbols, characters, terms, numbers or the like. It should be borne in mind, however, that all of these and similar terms are to be associated with the appropriate physical quantities and are merely convenient labels applied to these quantities. Unless specifically stated otherwise, it will be appreciated that throughout the description of the embodiments of the present invention, use of terms such as "processing", "computing", "calculating", "determining", "displaying" or the like, refer to the action and processes of a computer system, or similar electronic computing device, that manipulates and transforms data represented as physical (electronic) quantities within the computer system's registers and memories into other data similarly represented as physical quantities within the computer system memories or registers or other such information storage, transmission or display devices.

[0039] As indicated above, one embodiment of the present invention is instantiated in computer software, that is, computer readable instructions, which, when executed by one or more computer processors/systems, instruct the processors/systems to perform the designated actions. Such computer software may be resident in one or more

computer readable media, such as hard drives, CD-ROMs, DVD-ROMs, read-only memory, read-write memory and so on. Such software may be distributed on one or more of these media, or may be made available for download across one or more computer networks (e.g., the Internet). Thus, a machine-readable medium includes any mechanism that provides (i.e., stores and/or transmits) information in a form accessible by a machine (e.g., a computer, network device, personal digital assistant, manufacturing tool, any device with a set of one or more processors, etc.). For example, a machine-accessible medium includes recordable/non-recordable media (e.g., read only memory (ROM); random access memory (RAM); magnetic disk storage media; optical storage media; flash memory devices; etc.), as well as electrical, optical, acoustical or other form of propagated signals (e.g., carrier waves, infrared signals, digital signals). Regardless of the format, the computer programming, rendering and processing techniques discussed herein are simply examples of the types of programming, rendering and processing techniques that may be used to implement aspects of the various embodiments of the present invention. These examples should in no way limit the embodiments of the present invention, which is best understood with reference to the claims that follow this description.

[0040] Thus, an image processing device has been described. In the forgoing description, various specific values were given names, such as "image data" and "quartile pointer," and various specific modules, such as the "interpolation module" and "transfer function module" have been described. However, these names are merely to describe and illustrate various aspects of the embodiments of the present invention, and in no way limit the scope of the embodiments of the present invention. Furthermore, various

modules, such as the TFM 108 and the histogram module 110 in Figure 1, can be implemented as software or hardware modules, or without dividing their functionalities into modules at all. The embodiments of the present invention are not limited to any modular architecture either in software or in hardware, whether described above or not.